

(19) Japan Patent Office (JP)

(12) Japanese Unexamined Patent  
Application Publication (A)(11) Japanese Unexamined Patent  
Application Publication Number

Hei 5-37795

(43) Publication date: February 12, 1993

(51) Int. Cl.<sup>5</sup> Identification codes JPO file numberH 04 N 1/44  
1/00

B

2109-5C  
4226-5C

Request for examination Not yet requested Number of 1 (Total of 10 pages)

(21) Application number H3-216028  
(22) Date of application July 31, 1991

(71) Applicant Nippon Infralogic Co., Ltd.  
Higashi Nishinobashi Arrow Bldg.  
3-12-11 Higashi Nishinobashi, Chuo-ku, Tokyo

(72) Inventor Kineo Matsui  
5-57 Otsu-cho, Yokosuka-shi

(72) Inventor Yasuhiro Nakamura  
C-202 2-choma Hashirimizu, Yokosuka-shi

(72) Inventor Kiyoshi Tanaka  
1-22-6 Onuma Matsugaoka, Fujisawa-shi

(74) Agent Nobusuke Honjo, Patent Attorney

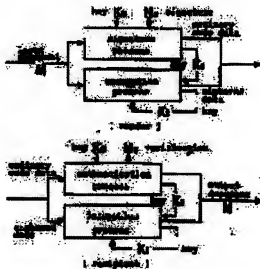
(54) [Title of the Invention] Facsimile

(57) [Abstract]

[Purpose] To implement signature and scramble processing by partially modifying and adding application software in the current card FAX system.

[Constitution]

It comprises of a means for embedding a signature sentence in a part of the image data in order to demonstrate legitimacy of the sender and the transmitted document, and a means for applying a scramble processing to the encoded image data.



A proposed confidentiality protected system

[Scope of the Claims]

[Claim 1]

A facsimile characterized in that it comprises of a means for embedding a signature sentence in a part of the image data in order to demonstrate legitimacy of the sender and the transmitted document, and a means for applying a scramble processing to the encoded image data.

[Detailed Description of the Invention]

[0001]

[Industrial Field of Application]

The present invention relates to a facsimile having confidentiality protection functions by signature and scrambling. In particular, it relates to a facsimile generally called a card FAX using a personal computer (herein after abbreviated as PC) as a communications means.

[0002]

[Prior Art]

A great revolution has occurred in the concept of document communications by the invention of facsimile communications. In a modern society requiring high speed, the facsimile has become an indispensable communications means<sup>(1)</sup>. However, along with increased values of utilization of facsimiles in document communications, the necessity of measures for confidentiality protection such as reliability of senders and document transmitted has also increased. That is, from general documents to important documents, confidentiality protection measures are required by the documents based on their purposes. In the current facsimile communications based on the CCITT standards, no measures have been taken regarding confidentiality protection in documents from facsimile terminals regarding transmitted data. The following operational problems have been pointed out: (1) error transfer due to dialing errors and danger of information leaks by a third party at the receiver side, (2) third party invasion without permission such as direct mail including advertisements and a variety of kinds of announcements cannot be avoided, and (3) there are no protections against wire tapping and altering on the communications network. Recently, special adaptors having authentication functions for senders and sent documents have been developed in order to prevent wire tapping and interception.

[0003] On the other hand, due to rapid progress in microelectronics technology, high functionality PCs can be used at low cost and attempts to use PCs that have been used independently as communications means as well have been made actively. This trend has been adapted due to the reasons that it is more economical than purchasing and utilizing a special communications terminal separately and that data processing and storage functions of PCs can be used additively. Facsimile communications using PCs have already been used in practice as in the use of extension board type communications adaptors called a card FAX (or PC FAX) and its effectiveness has been demonstrated.

[0004] Now, Figure 1 shows an example configuration of the conventional card FAX. A card FAX is the most compact communications adaptor that enables facsimile communications immediately if it is inserted through the extension slot of a PC to be connected to the telephone line. In terms of hardware, a NCU (network control unit) and a modem for control signals and image signals are loaded on an extension board which functions as a communications interface. On the other hand, attached application software generally implements transfer control

and encoding functions of image data. In the case of a card FAX, the software also functions to convert text data to image data and encoded data so that the documents prepared by word processing programs can be sent directly by facsimile communications. In this report, we focused on the greatest advantage of facsimile communications using a PC, that is, the degree of freedom of additively implementing various functions using the software, and attempted to have security functions with the card FAX. In addition to the conventional general functions such as sending/receiving, reading the document, converting files, and outputting documents, the present invention implements a signature showing legitimacy of the sender and the transmitted document, and a scramble function for the purpose of confidentiality of the document by using software.

[0005]

[Problems to be Solved by the Invention] On the assumption that a card FAX using PCs will become a powerful communications means in the future, a card FAX having security functions will be required. The purpose of the present invention can be implemented by only partially modifying and adding application software in the current card FAX system and signatures and scrambling can be implemented by the software using the processing functions of a microprocessor.

[0006]

[Means for Solving the Problems] To solve the aforementioned problems, the facsimile that the present invention provides comprises of a means for embedding a signature sentence in a part of the image data in order to demonstrate legitimacy of the sender and the transmitted document, and a means for applying a scramble processing to the encoded image data.

[0007]

[Examples] Figure 2 is a block diagram showing a confidentiality protection system in the present invention's facsimile. The sender side consists of a signature process and an encoding process, and the receiver side consists of an authentication process and a decoding process.

[0008] In the signature process at the sender side, a signature is applied to a part of the input document in order to demonstrate legitimacy of the sender and the transmitting document. That is, for a part of the document image data stored in the memory, for example, for the area where the name and address of the sender is stored, a signature sentence is embedded by the processing functions of the microprocessor and encoded to be transmitted. The encoded data of a part of the document where the signature is embedded is directly sent without scrambling. Thus, the document header (header part) can be read as normal. In this process, the sender inputs a signature ciphering key  $K_0$  and signature sentence  $N_s$  along with the transmitting document  $M_1$ . When the signature process (embedding the signature sentence) is completed, a first scramble key  $K_1$  in the encryption process is generated to shift to the encryption process.

[0009] In the encryption process, a scrambling process is applied to the image data encoded by the key  $K_i$  ( $i = 1, 2, \dots$ ) generated from the sequential image data. A method of scrambling will be described later. The encoded data (binary) generated by blocking by the unit of power of 2 are followed by transposition encryption to be transmitted.

[0010] On the other hand, in the authentication process at the receiver side, a signature ciphering key  $K_0$  common to the sender and the receiver is input and the sender and the transmitted document are confirmed by legitimacy of

the signature sentence  $N_s$  extracted from the received data  $F(M_i)$ . If the signature sentence  $N_s$  is a data having a meaning, the legitimacy of the sender and the transmitted document are confirmed and shifts to the decoding process based on the decoding key  $K_1$  generated. In contrast, if  $N_s$  is not a data having a meaning, the sender or the transmitted document is determined to be not legitimate and the subsequent processing is interrupted and the received document is aborted.

[0011] In the decoding process, the encrypted received data is reverse-scrambled in blocks as units based on the key  $K_i$  ( $i=1, 2, \dots$ ) generated from the sequentially decoded image data so that the encoded data are transformed to image data.

[0012] In the proposed system as described above, only the sender and the receiver cipher manage the common key  $K_0$ . In addition, the scramble (reverse-scramble) key  $K_i$  ( $i=1, 2, \dots$ ) are sequentially generated during the processing process depending upon the structure of the document image so that it is difficult for the third party who does not know the signature, the structure of the encryption function, and the cipher key  $K_0$  to detect this.

[0013]

[Signature method]

[Embedding of a signature sentence] The signature indicating the legitimacy of the sender and the transmitting document is carried out by directly embedding the signature sentence into the digitalized document image data. That is, from the image data resolved into the scanning lines, the scanning lines designated randomly by the keys are referred and the oddness/evenness of the distance among the variable pixels on the reference scanning lines and encoded scanning lines are modulated by 1 bit of signature data.

[0014] Initially, variable pixels on the encoded scanning lines and  $s$  pieces of reference scanning lines are defined as follows (See Fig. 3).

$a_1$ : First variable pixel of the target run length (hereinafter referred to as RL) on the encoded scanning lines. In this case, No. 1 RL is excluded.

$b_1(j)$ : Same color (as  $a_1$ ) variable pixel to corresponding to  $a_1$  on the  $j$ -th ( $1 \leq j \leq s$ ) reference scanning line. That is, if the pixel immediately above  $a_1$  is in the same color as  $a_1$ , the first pixel of the previous RL, if the pixel immediately above  $a_1$  is in an opposite color from  $a_1$ , the first pixel of the RL located right of  $a_1$ .

$\Delta_j$  ( $a_1 \text{ } b_m$ ): a distance between the variable pixel  $a_1$  and  $b_m$  ( $j$ )

$O_j$ : indicates even/oddness of  $\Delta_j$  ( $a_1 \text{ } b_m$ ). 0, if  $\Delta_j$  ( $a_1 \text{ } b_1$ ) is an even number and 1, if  $\Delta_j$  ( $a_1 \text{ } b_1$ ) is an odd number.

[0015] In the signature keys of  $r$  bits:

$K_0 = \{k_0(t_0) \mid t_0 = 0, 1, \dots, r-1\}$ ;

$$k_0(t_0) = 0, 1 \quad (1)$$

if  $k_0(t_0) = 1$ , a length of the bit from immediately before  $k_0(t_0') = 1$  to  $k_0(t_0) = 1$

$$j = t_0 - t_0' \quad (2)$$

(where at first  $k_0(t_0) = 1$ , a distance between  $j = t_1$  and variable pixel  $b_1(j)$  on the  $j$ -th reference scanning line is  $\Delta_j$   $0 + 1$ ). Then, 1 bit signature data  $n_s(u)$  extracted from  $q$  bit signature sentence

$$N_s = \{n_s(u) \mid u=0, 1, \dots, q-1; n_s(u) = 0, 1\} \quad (3)$$

if embedded in the even/oddness  $\emptyset_j$  of the variable pixel a (a1 b1) on the encoded scanning lines by the following procedures.

- (1) If  $\Delta_j(a1\ b1) > 0$ ;  $\Delta_j(b0\ a1) \geq 1$  and  $\Delta_j(b1\ a2) \geq 1$  are satisfied,
  - (a) Directly if  $\emptyset_j \times ORns(u) = 0$
  - (b) Shifts to the right by the portion of one pixel if  $\emptyset_j \times ORns(u) = 1$
- (2) If  $\Delta_j(a1\ b1) \leq 0$ ;  $\Delta_j(b0\ a1) \geq \Delta_j(a1\ b1) + 2$  is satisfied,
  - (a) Directly if  $\emptyset_j \times ORns(u) = 0$
  - (b) Shifts to the left by the portion of one pixel if  $\emptyset_j \times ORns(u) = 1$

However, if  $k0(t0) = 0$ , signature data is not embedded.

[0016] The signature data embedded in these procedures can be easily decoded in the following procedures:

- (1) If  $\Delta_j(a1\ b1) > 0$ ;  $\Delta_j(b0\ a1) \geq 1$  and  $\Delta_j(b1\ a2) \geq 1$  are satisfied,  
 $ns(u) = \emptyset_j$
- (2) If  $\Delta_j(a1\ b1) \leq 0$ ;  $\Delta_j(b0\ a1) \geq \Delta_j(a1\ b1) + 2$  is satisfied,  
 $ns(u) = \emptyset_j$

And then, the extracted 1 bit signature data  $ns(u)$  is assembled as a signature sentence  $Ns$ . The aforementioned regulations (1) and (2) are given in Equation (5).

[0017] Since the signature sentence is embedded depending upon the structure of the document image (RL pattern over several scanning lines), it is difficult for the third party to forge or alter the document image  $Mf$ .

[0018]

[Generation of scramble keys] Next, a method of generating the scramble keys  $K_i$  ( $i = 1, 2, \dots$ ) in the encryption process will be described below.  $K_i$ , based on  $K_{i-1}$  used immediately before (initial signature key  $k0$ ), is generated from the structure of the document image data. Initially, regarding the maximum number of reference scanning lines  $s$ , a minimum integer value  $h$  satisfying

$$s \leq 2^h \quad (6)$$

is calculated and given as  $h0$ .

From the bit series of the immediate key  $K_{i-1}$ , next  $h0$  pieces of bits

$$\{k_{i-1}((ti-1 + \alpha) \bmod r) \mid \alpha = 0, 1, \dots, h0-1\} \quad (7)$$

are extracted and then transformed to decimal  $d0$ . Next, according to

$$l = (d0 + 1) \bmod s \quad (8)$$

a reference scanning line number  $l$  is determined. The even/oddness  $\emptyset_l$  of the distance  $\Delta_l(a1\ b1)$  between the variable pixel  $a1$  on the encoded scanning lines and the variable pixel  $b1(l)$  is determined and then  $kl(ti) = \emptyset_l$  is given. This operation is continuously repeated  $r$  times to obtain scramble keys

$$K_i = \{k_i(ti) \mid ti = 0, 1, \dots, r-1; K_i(ti) = 0, 1\} \quad (9)$$

[0019] In this report, the scramble method wherein the data encoded by sequentially generated keys depending upon the structure of the document image is transpositioned is used. An actual method will be shown below.

[0020]

[Scramble method]

[Concept of encryption] An encrypted sentence obtained by encryption of a plain sentence M by the key K is expressed as

$$C = E(K, M) \quad (10)$$

In the transposition method in the conventional hardware, the structure of E is fixed so that the degree of freedom of transposition is fully dependent upon the degree of freedom of the key

$$C_i = D(K_i, M) \quad (i = 1, 2, \dots) \quad (11)$$

was obtained. In contrast, according to the present method, E is configured by a software system. If the key  $K_i$  is input, the system automatically generates and orients different partial encrypter (decoder)  $E_{ij}$  and partial key  $K_{ij}$  ( $i = 1, 2, \dots; j = 1, 2, \dots, n$ ) and they are further combined to output

$$C = E_{i1}(K_{i1}, E_{i2}(K_{i2}, \dots, E_{in}(K_{in}, M))) \dots \quad (12)$$

[0021] In the present system, Equation (12) is implemented by the following procedures. Initially, when the scramble keys  $K_i$  ( $i = 1, 2, \dots$ ) are input,  $E_{ij}$  and  $K_{ji}$  are developed in the memory using a portion of  $K$ 's. As shown in Fig. 4, a portion of the bit series of keys  $K_i$  is used as a machine language program and the subsequent series are used as a partial key. Next, the encoded data stored on the memory is blocked by units by powers of 2, and using the program generated, scrambling is executed using the coordinate values of the individual data in the blocks as arguments. Using the operation results as new coordinate values, the encoded data are output. After transposition of one block is completed, a scramble key  $K_{i+1}$  for the next block is output along with the encrypted series and the same processing is repeated based on  $K_{i+1}$ . The flow of this process is shown in Fig. 5.

[0022] Each of  $E_{ij}$  used must be a mapping function of bijection (1:1) due to the necessity of configuring the block transposition encryption on the whole. Therefore, in the present system, basic operation instruction and the extended operation of the central processing unit (CPU) are selected as  $E_{ij}$ . As a result, an encrypter and a decoder can perform a high speed transposition processing using only a small amount of memory

[0023]

[CPU instruction to become a bijection mapping function]

Among the instructions of the CPU such as addition/subtraction/multiplication/division, those configuring a bijection mapping between the numbers operated and the operation results are limited. The CPU instructions used are listed below. In this case, the length of the register of the CPU generating the aforementioned scrambler is expressed as R. Therefore, the addressing range in this register is  $0 \sim 2R-1$  and a one block to be input has this size as a unit.

[0024]

[Exclusive OR]

A processing of performing an exclusive OR operation for the constant (encryption key) provided for each bit and every bit in the register is a bijection. If this operation is expressed by XOR, number to be operated is M, and the key constant is  $K_{XOR}$ , the operation results are expressed by

$$C = XOR(K_{XOR}, M) \quad (13)$$

Obviously,

$$M = \text{XOR}(K_{\text{XOR}}, C) \quad (14)$$

In this operation, respective data having two coordinate values M and C with the relationships expressed by Equation (13) or Equation (14) are replaced by each other. That is, all the data in the block are substituted by the pairs satisfying the aforementioned relationships.

[0025]

[Left/Right Rotation]

The bits in the register are shifted by the specified number of bits in the specified direction, and the places expelled out of the register are input from the opposite side, in an operation called a rotation operation that is a bijection. The left rotation is expressed by ROL and right rotation is expressed by ROR.

[0026]

[Addition/subtraction]

A processing of adding and subtracting the data in the register is also a bijection. The addition is expressed by ADD and the subtraction is expressed by SUB. Generally in an arithmetic operation instruction such as ADD, when the results are overflowed from the places, flag registers are affected. However, only the data in the register is considered by ignoring this point,

$$C = \text{ADD}(K_{\text{ADD}}, M) = M + K_{\text{ADD}} \bmod 2^R \quad (15)$$

That is, individual data are shifted in parallel for a distance of  $K_{\text{ADD}}$ . In addition to the aforementioned CPU instructions constituting the bijective mapping, the following extension operation becomes also a bijective mapping and improves the transposition effect.

[0027]

[Replacement of bit positions]

A processing of replacing the bits in the register to specific positions is also bijection. This operation is expressed by EXT1 below. The keys  $K_{\text{EXT1}}(i)$  ( $i = 0, 1, \dots, R-1$ ) are an array consisting of R pieces of elements corresponding to the length of register R and the respective values indicate positions of bits when they are shifted.

[0028]

[Positions in each subspace]

The entire address space expressed by the length of register R is partitioned into subspaces and when a bijection is operated in each subspace, the entire transposition is a bijection. As a method of partitioning into subspaces, there is a method of shortening the register length, and a method of classification by the pieces of each bit included in the register. Here, we used subspaces in the latter group. The operable ROL is applied in the subspaces and expressed as EXT2. However, it is applied to the subspaces excluding the cases when all bits in the register are bit 1 or 0.

Thus, the keys  $K_{\text{EXT2}}(i)$  ( $i = 0, 1, \dots, R-2$ ) is an array consisting of R-1 pieces of elements where R is the length of the register, and the first element indicates the key ROL which is applied to the subspace where i pieces of bits 1 are present in the register.

[0029]

[Encrypter/decrypter]

The encrypter and the descriptor given by Equation (12) generate machine language instructions and their operands from the bit series of keys  $K_i$  ( $i = 1, 2, \dots$ ). 2 or 3 bits are input from  $K_i$  and  $n$ -step of a group of instructions are generated based on the transformation table shown in Table 1 and developed as a series of partial encrypters  $E_{ij}$  ( $j = 1, 2, \dots, n$ ) in the coded domain of the main memory. In this case, an operand required in each transposition operation, that is partial keys  $K_{ji}$  ( $j = 1, 2, \dots, n$ ) is obtained from the bit series of  $K_i$ . However, the number of steps  $n$  of the encrypter is set in advance. For example, if the length of register is  $R = 16$  bits, the number of bits of partial keys required in each operation are as shown in Table 2.

[Table 1]

Transformation table

Input bits	Output codes
110	XOR
000	ROL
001	ROR
010	ADD
011	SUB
111	EXT1
10	EXT2

[Table 2]

Number of bits of partial keys required for operation

Codes	Number of bits required
XOR	16
ROL	4
ROR	4
ADD	16
SUB	16
EXT1	64 (= 4 x 16)
EXT2	60 (= 4 x 15)

[0030]

[Actual procedures for signature and encryption] In the present example, the confidential data between the sender and the receiver is  $(s, n, K_0)$ . The actual procedures are described below.

[0031]

[Signature process] Step 1: Enter a document image  $M_i$ , a signature key  $K_0$  and a signature sentence  $N_s$  while keeping  $t_0 \leftarrow 0$ ,  $t_1 \leftarrow 0$ ,  $u \leftarrow 0$ .

Step 2:  $s$  pieces of scanning lines are stored in the memory and encoded and transmitted.

Step 3: The processes from Step 4 to Step 9 are repeated until completing embedding of the signature sentence  $N_s$ .

Step 4: The processes from Step 5 to Step 8 are repeated until completing scanning of the encoding scanning lines.

Step 5: Detects a variable pixel  $a_1$ .

Step 6:  $h_0$  bits containing  $k_0$  ( $t_0$ ) are extracted from  $K_0$  and converted to decimal  $d_0$ , and  $l$  is decided from Equation (8). A variable pixel  $b_1^{(0)}$  on the first reference scanning line is detected and  $k_1$  ( $t_1$ )  $\leftarrow 0$  is given.  $t_l \leftarrow t_l + 1$  is also given.

Step 7: If  $k_0(t_0) = 1$ , a length of bit  $j$  from the immediately before  $k_0(t_0') = 1$  to  $k_0(t_0) = 1$  is determined by Equation (2). A variable pixel  $b^{(0)}$  on the  $j$ -th reference scanning line is detected and the signature data  $ns(u)$  is embedded by Equation (4). When  $ns(u)$  is embedded,  $u \leftarrow u + 1$  is given.

Step 8:  $t_0 \leftarrow (t_0 + 1) \bmod r$  is given.

Step 9: The encoded scanning lines are encoded and transmitted as well as the encoded scanning lines are updated to the first reference scanning line and the  $v$ -th reference scanning line is updated to the  $v+1$  reference scanning line ( $v = 1, 2, \dots, s-1$ ).

Step 10: Shifting to the encryption process.

[0032]

[Encryption process] Step 1: Select  $i = 1$ , input  $n$ .

Step 2: Step 3 to Step 8 are repeated until all encoded data are completed.

Step 3: Set  $t_i \leftarrow 0$  and  $t_{i+1} \leftarrow 0$ .

Step 4: Step 5 to Step 8 are repeated until the amount of encoded data exceeds  $2^R$  bits or more.

Step 5: Step 6 to Step 7 are repeated until scanning of the encoded scanning lines are completed.

Step 6: A variable pixel  $a$  is detected.

Step 7:  $h_0$  bits containing  $k_i(t_i)$  are extracted from  $K_i$  and converted to decimal  $d_0$ , and  $l$  is decided from Equation (8). A variable pixel  $b^{(0)}$  on the first reference scanning line is detected and  $k_{i+1}(t_{i+1}) \leftarrow 0$  is given.  $t_i = (t_i + 1) \bmod r$ ,  $t_{i+1} = (t_{i+1} + 1) \bmod r$  are given.

Step 8: The encoded scanning lines are encoded as well as the encoded scanning lines are updated to the first reference scanning line and the  $v$ -th reference scanning line is updated to the  $v+1$  reference scanning line ( $v = 1, 2, \dots, s-1$ ).

Step 9:  $n$ -step transposition crypter is developed from  $K_i$  to the encoded domain of the main memory.

Step 10: The coded data at the address  $m (= 0 \sim 2^R - 1)$  are transpositioned at the address  $m' (= 0 \sim 2^R - 1)$  on the buffer memory provided separately by the generated crypter, and then output and transmitted from the address 0 in the order.  $i \leftarrow i + 1$ .

[0033]

[Authentication process] Step 1: Enter received data  $F(M_i)$  and authentication key  $K_0$ , while keeping  $t_0 \leftarrow 0$ ,  $t_1 \leftarrow 0$ ,  $u \leftarrow 0$ .

Step 2: Decode the encoded data for the portion of  $s$  scanned lines and store them in the memory to be used as reference scanning lines.

Step 3: Step 4 to Step 10 are repeated until decoding of signature  $N_s$  is completed.

Step 4: Encoded data is decoded for the portion of  $l$  scanning line to use as encoded scanning line.

Step 5: Step 6 to Step 9 are repeated until scanning of the encoded scanning lines is completed.

Step 6: A variable pixel  $a$  is detected.

Step 7:  $h_0$  bits containing  $k_0(t_0)$  are extracted from  $K_0$  and converted to decimal  $d_0$ , and  $l$  is decided from Equation (8). A variable pixel  $b^{(0)}$  on the first reference scanning line is detected and  $k_1(t_1) \leftarrow 0$  is given.  $t_1 \leftarrow t_1 + 1$  is also given.

Step 8: If  $k_0(t_0) = 1$ , a length of bit  $j$  from the immediately before  $k_0(t_0') = 1$  to  $k_0(t_0) = 1$  is determined by Equation (2). A variable pixel  $b^{(0)}$  on the  $j$ -th reference scanning line is detected by Equation (5) and the signature data  $ns(u)$  is extracted. When  $ns(u)$  is extracted,  $u \leftarrow u + 1$  is given.

Step 9:  $t_0 \leftarrow (t_0 + 1) \bmod r$  is given.

Step 10: The encoded scanning lines are updated to the first reference scanning line and the  $v$ -th reference scanning line is updated to the  $v+1$  reference scanning line ( $v = 1, 2, \dots, s-1$ ).

Step 11: If  $N_s$  is legitimate, shifting to the decoding process, otherwise, the received data  $F(M_f)$  is aborted.

[0034]

[Decoding process] Step 1: Select  $i = 1$ , input  $n$ .

Step 2: Step 3 to Step 8 are repeated until all encoded data are completed.

Step 3:  $2^R$  bits are extracted from the received data  $F(M_i)$  and set  $t_i \leftarrow 0$  and  $t_{i+1} \leftarrow 0$ .

Step 4: A  $n$ -step transposition encrypter is developed from  $K_i$  to the encoded domain of the main memory.

Step 5: The received data at the address  $m (= 0 \sim 2^R - 1)$  are transpositioned at the address  $m' (= 0 \sim 2^R - 1)$  on the buffer memory provided separately by the generated decoder, and then output from the address 0 in the order.  $i \leftarrow i + 1$ .

Step 6: The decoded data are converted to image data and the Step 7 to Step 10 are repeated for the scanning lines converted.

Step 7: Step 8 to Step 9 are repeated until scanning of the encoded scanning lines is completed.

Step 8: A variable pixel  $a_l$  is detected.

Step 9:  $h_0$  bits containing  $ki(t_i)$  are extracted from  $K_i$  and converted to decimal  $d_0$ , and  $l$  is decided from Equation (8). A variable pixel  $b^{(0)}$  on the first reference scanning line is detected and  $ki+1(t_{i+1}) \leftarrow 0$  is given.  $t_i = (t_i + 1) \bmod r$ ,  $t_{i+1} = (t_{i+1} + 1) \bmod r$  are given.

Step 10: The encoded scanning lines are updated to the first reference scanning line and the  $v$ -th reference scanning line is updated to the  $v+1$  reference scanning line ( $v = 1, 2, \dots, s-1$ ).

[0035]

[Investigating the scrambling effect]

In order to investigate whether the 0.1 bit series of encoded data output as the result of the execution of the said scrambling procedures are sufficiently random, with respect to the number series of the  $M$  series by the feedback shift resistor that has been proven to have sufficient pseudo-randomization, the properties were compared. However, a MH system was used as an encoding method for the facsimile document, and CCITT Test Chart No. 4 was used as an input document.

[0036] (1) Probability of occurrence of sequential pieces

"Sequential" means a line segment consisting of  $p$  sequential pieces of 1 or 0 in the binary series and both ends are either 0 or 1. In the  $M$  series, with respect to the 1 or 0 sequential pieces, the following formula (16) is expressed:

$$(\text{Sequential pieces with a length } p) / (\text{Sequential pieces with a length } p + 1) = 2 \quad (16)$$

On the  $q$ -step shift register, this condition is satisfied except for the cases when a length is  $q$  and  $q-1$ . The sequential status was investigated for the encoded data of the MH system and when the data were scrambled with a register

length R of 16 bits with a basic instruction steps n of 32. Figure 6 shows the results of logarithmic graphs of the sequential pieces plotted based on log 2. The dotted line indicates the result for the equivalent steps (16 steps) of the shift register. According to this result, the data series scrambled by the present method demonstrated excellent results in the case of a shift register with respect to the sequential pieces. If n increased, better results were confirmed to be obtained.

[0037] (2) Autocorrelation function

Next, randomized properties of the scrambled data series were investigated using the discrete type autocorrelation function defined by the following equation<sup>(11)</sup>

[Formula 1]

$$C_0(h) = \lim_{j \rightarrow \infty} \frac{1}{2j+1} \sum_{r=-j}^{r=j} y_r y_{r+h} \quad (17)$$

where  $\{y_r\}$  indicates a term in the binary series, and h is a distance between the two terms  $\{y_r\}$  and  $\{y_{r+h}\}$

if  $h \neq 0$ ,  $C_0(h) = 0$

if  $h = 0$ ,  $C_0(h) = 1$  (18)

The randomized properties of the series can be characterized as shown above. The respective autocorrelation functions were obtained for the encoded data of the MH system and when the data were scrambled with a register length R of 16 bits with a basic instruction steps n = 32. The results are shown in Fig. 7. The operation results of the present method were found to satisfy Equation (18) similarly in the case of the autocorrelation function of the shift register as indicated by the dotted line.

[0038]

[Effects of the Invention]

As was explained in detail in the aforementioned example, in the present facsimile system, (1) a signature is applied by embedding a signature sentence indicating legality of the sender and the transmitted document in a part of the image data, and (2) a scramble processing is applied to the encoded image data. By employing these methods, an extension board type FAX adaptor implemented by the present invention is simple in terms of hardware so that it is available at a low cost. Therefore, when compared to an adaptor specialized for confidentiality protection, security communications can be implemented at a low cost. In addition, the scrambling processing is carried out by blocking the encoded data in units of powers of two for transposition ciphering so that the advantage is that data is not limited by various encoding methods such as MH, MR and MMR for facsimiles and the quantity of transmission codes. In addition, the present invention's facsimile system can be implemented only by partially modifying and adding to the application software used in the currently used card FAX system. As a result, security communications can be implemented at a lower cost than a special adaptor for confidentiality protection.

[Brief Explanation of the Drawings]

[Fig. 1] A diagram showing the configuration example of the conventional card type FAX.

[Fig. 2] A diagram showing the confidentiality protection system as an example of the present invention.

[Fig. 3] A diagram showing definitions of the variable pixels on the encoded scanning lines and s pieces of reference scanning lines.

[Fig. 4] A diagram showing the partial encrypters (decrypters)  $E_{ij}$  and partial keys  $K_{ij}$ .

[Fig. 5] A diagram showing a flow of the scrambling processing in the present invention's example.

[Fig. 6] Logarithmic graphs plotted based on  $\log 2$  for the sequential pieces in the example of the present invention.

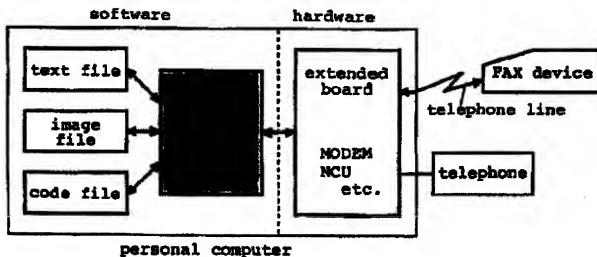
[Fig. 7] A diagram showing autocorrelation functions.

[Fig. 4]



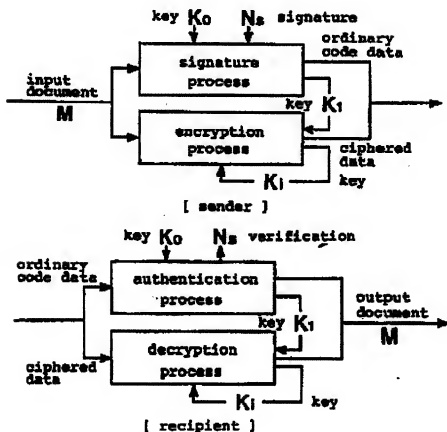
Procedures for generating  $E_{ij}$  and  $K_{ij}$

[Fig. 1]



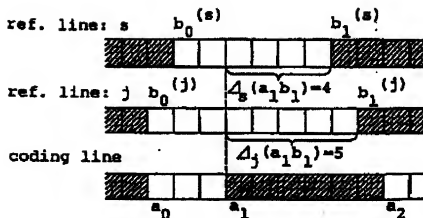
Configuration of a card type FAX

[Fig. 2]



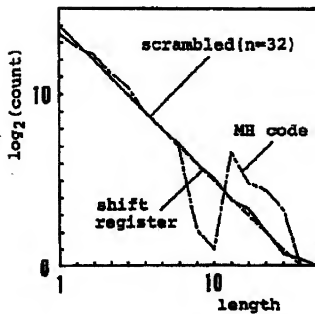
A proposed confidentiality protected system

[Fig. 3]



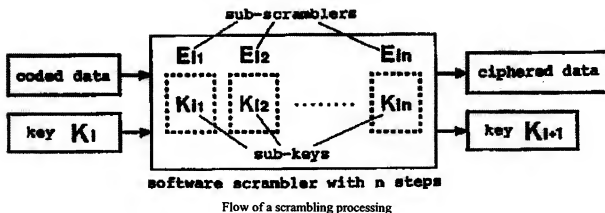
Definitions of variable pixels

[Fig. 6]

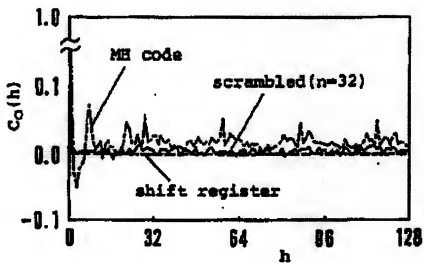


Sequential pieces

[Fig. 5]



[Fig. 7]



Autocorrelation functions

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-037795

(43)Date of publication of application : 12.02.1993

(51)Int.Cl.

H04N 1/44

H04N 1/00

(21)Application number : 03-216028

(71)Applicant : NIPPON INFURAROJITSUKU KK

(22)Date of filing : 31.07.1991

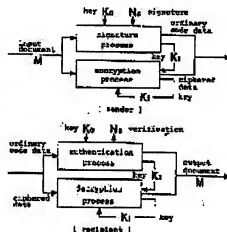
(72)Inventor : MATSUI KINEO  
NAKAMURA YASUHIRO  
TANAKA KIYOSHI

## (54) FACSIMILE EQUIPMENT

(57)Abstract:

**PURPOSE:** To reduce the cost of an expansion board type FAX adaptor by imbedding a signature of a sender and a transmission document into a picture data and applying scramble processing to a coded picture data.

**CONSTITUTION:** A signature sentence is imbedded to part of a document picture data by microprocessor processing. In this case, a signature ciphering key K0 and a signature sentence Ns are inputted together with a transmission document M to generate a scramble key K1 thereby setting the mode to the ciphering process. A picture data coded by using the key K1 is sent while being transposition ciphering as two binary data subjected to block processing in the unit of power. A receiver side receives a key K0 in common to a sender and a receiver and the transmission document is confirmed based on the adequacy of the signature sentence Ns. When the adequacy is confirmed, the key K1 is used to set the decoding process. When not adequate, it is aborted. Thus, since only the sender and the receiver cipher and manage the key K0, a 3rd party cannot detect the key.



特開平5-37795

(43)公開日 平成5年(1993)2月12日

(51)Int.Cl. <sup>5</sup>	識別記号	庁内整理番号	F I	技術表示箇所
H 0 4 N 1/44		2109-5C		
1/00	B	4226-5C		

審査請求 未請求 請求項の数1(全10頁)

(21)出願番号 特願平3-216028

(22)出願日 平成3年(1991)7月31日

特許法第30条第1項適用申請有り 1991年1月31日～2月2日 電子情報通信学会情報セキュリティ研究専門委員会主催の「1991年暗号と情報セキュリティシンポジウム」において文書をもって発表

(71)出願人 391053319

日本インフラロジック株式会社  
東京都中央区東日本橋3-12-11 東日本橋アロービル5階

(72)発明者 松井 甲子雄

横須賀市大津町5-57

(72)発明者 中村 康弘

横須賀市走水2丁目無香地C-202

(72)発明者 田中 清

藤沢市鶴沼松が岡1-22-6

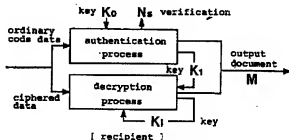
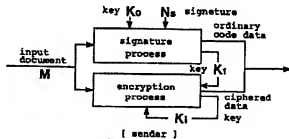
(74)代理人 弁理士 本庄 伸介

(54)【発明の名称】 ファクシミリ

(57)【要約】

【目的】 現行のカードFAXシステムにおけるアプリケーション・ソフトウェアを部分的に変更し、追加するだけで署名およびスクランブルを実現する。

【構成】 送信者および送信文書の正当性を示すための署名文を画像データの一部に埋め込む手段と、符号化された画像データにスクランブル処理を施す手段とを含む。



提案する機密保護システム

## 【特許請求の範囲】

【請求項1】 送信者および送信文書の正当性を示すための署名文を画像データの一部に埋め込む手段と、符号化された画像データにスクランブル処理を施す手段とを含むことを特徴とするファクシミリ。

## 【発明の詳細な説明】

## 【0001】

【産業上の利用分野】 本発明は、署名およびスクランブルによる機密保護機能を備えるファクシミリに関し、特にパーソナルコンピュータ（以下パソコンと略称する）による通信手段とし、カードFAXと通称されるファクシミリに関する。

## 【0002】

【従来の技術】 ファクシミリ通信により文書通信の概念は大きく変革され、即時性を求められる現在社会においてファクシミリはもはや不可欠な通信手段となっている。<sup>(1)</sup> しかし、文書通信におけるファクシミリの利用価値が高まるに伴い、送信者や送信文書の信頼性などの機密保護に関する対策の必要性もまた増加している。すなわち、一般文書から重要文書に到るまで、その使用目的に応じた文書の機密保護対策が必要とされている。CITTの標準化による現行ファクシミリ通信では、ファクシミリ端末から端末間の送信データに到るまで文書の機密保護に関する対策が何も施されていないために、

(1) ダイヤルミスなどの誤伝送や受信端末における第三者の介入による情報漏洩の危険性がある、(2) 広告や各課案内などのダイレクトメール的な第三者の無断侵入を防ぐことができない、(3) 通信ネットワーク上での盗聴・改ざんに対して無防備である、などの運用上の問題点が指摘されている。そこで、端末間の盗聴・傍受を防止し、送信者や送信文書の認証機能を持つ専用アダプタが近年いくつか開発されている。

【0003】 一方、マイクロエレクトロニクス技術の急激な進歩により最近では高機能なパソコンを低価格で利用できるようになり、従来独立的に利用されていたパソコンを通信手段として利用する試みが活発化している。このような傾向は、専用の通信端末を別途購入・利用するより経済的であり、さらに、パソコンのデータ処理・蓄積機能を付加的に利用することなどの理由による。パソコンを用いたファクシミリ通信についても、カードFAX（またはパソコンFAX）と称する拡張ボード型の通信用アダプタがすでに実用化され、その有効性が示されている。

【0004】 まず、従来のカードFAXの構成例を図1に示す。カードFAXは、パソコンの拡張スロットに挿入して電話回線に接続すれば直ちにファクシミリ通信を行うことができる最もコンパクトな通信アダプタである。ハードウェア的には、拡張基板上にNCU（ネットワーク制御装置）および制御番号用と画像番号用のMODEMが搭載されており、通信インタフェースとして機能

する。一方、付属のアプリケーション・ソフトウェアは伝送制御および画像データの符号化機能を実現するのが一般的である。また、カードFAXではフープロなどで作成した文書を直接ファクシミリ通信するために、テキストデータを画像データや符号データへ変換する機能もソフトウェアで行う。そこで、この報告ではパソコンによるファクシミリ通信の最大の利点、すなわち、ソフトウェア次第で様々な機能を付加的に実現できる自由度に着目し、カードFAXにセキュリティ機能を持たせることを試みる。従来の送・受信、文書読み取り、ファイル変換や文書出力などの一般的な機能の他に、送信者と送信文書の正当性を示すための署名および文書の秘匿を目的とするスクランブル機能をソフトウェアにより実現し、付加する。

## 【0005】

【発明が解決しようとする課題】 そこで、この発明では、将来パソコンを利用したカードFAXが有力な通信手段となることを想定し、セキュリティ機能を有するカードFAXが求められる。本発明の目的は、現行のカードFAXシステムにおけるアプリケーション・ソフトウェアを一部変更・追加するだけで実現可能であり、マイクロプロセッサの処理機能を利用してソフトウェアにより署名およびスクランブルを実現することにある。

## 【0006】

【課題を解決するための手段】 前述の課題を解決するために本発明が提供するファクシミリは、送信者および送信文書の正当性を示すための署名文を画像データの一部に埋め込む手段と、符号化された画像データにスクランブル処理を施す手段とを含むのである。

## 【0007】

【実施例】 図2は本発明のファクシミリにおける機密保護システムを示すブロック図である。送信側では署名プロセスおよび暗号化プロセス、受信側では認証プロセスおよび復号プロセスのそれぞれ2つのプロセスから構成される。

【0008】 まず、送信側における署名プロセスでは、送信者と送信文書の正当性を示すために入力文書の一部に対して署名を施す。すなわち、メモリに蓄積された文書画像データの一部、たとえば、送受信者の名前および宛先などが記載されている領域に対して、マイクロプロセッサの処理機能により署名文を埋め込み、符号化した後送信する。署名文の埋め込まれた文書画像の一部の符号データは、スクランブルせずにそのまま伝送する。よって、文書冒頭部（ヘッダ部）は従来通り判読できるものである。送信者は、このプロセスにおいて送信文書Miと共に署名用の秘匿鍵K0および署名文N0を入力する。署名処理（署名文の埋め込み）が終了すると、つぎの暗号化プロセスにおける当初のスクランブル鍵K1を生成し、暗号化プロセスに移行する。

【0009】 暗号化プロセスでは、逐次画像データから

生成される鍵 $K_i$  ( $i=1, 2, \dots$ )により符号化される画像データをスクランブル処理する。スクランブル方法については後に詳述するが、2のべき乗単位にブロック化した符号(2道)データを転置暗号化し、伝送する。

【0010】一方、受信側における認証プロセスでは、送信者と受信者の共通の秘密鍵 $K_0$ が入力され、受信データ $F(M_i)$ より抽出された署名文 $N_s$ の正当性により送信者と送信文書の確認が行われる。このとき、もし、署名文 $N_s$ が意味のあるデータならば送信者および送信文書の正当性が認められ、生成された復号鍵 $K_i$ をもとに復号プロセスに移行する。逆に、 $N_s$ が意味のないデータならば送信者または送信文書は正当でないと判断され、以下の処理を中止して受信文書を棄却する。

【0011】復号プロセスでは、逐次復号された画像データから生成される鍵 $K_i$  ( $i=1, 2, \dots$ )により暗号化された受信データをブロック単位に逆スクランブルし、符号データから画像データへと変換する。

【0012】このように提案するシステムでは、送信者および受信者は共通鍵 $K_0$ だけを秘密・管理すればよい。また、スクランブル(逆スクランブル)鍵 $K_i$  ( $i=1, 2, \dots$ )は文書画像の構造に依存して処理過程において逐次生成されるので、署名、暗号化関数の構造および秘密鍵 $K_0$ を知らない第三者がこれらを検出することは困難であるという特徴がある。

$$K_0 = \{k_0(t_0) \mid t_0 = 0, 1, \dots, r-1; k_0(t_0) = 0, 1\} \quad (1)$$

において $k_0(t_0) = 1$ のとき、直前の $k_j = t_0 - t_0'$

を求める(ただし、最初の $k_0(t_0) = 1$ では $j = t_0 - t_0' + 1$ とする)。そして、符号化走査線上的変化要素 $a$

$$N_s = \{n_s(u) \mid u = 0, 1, \dots, q-1; n_s(u) = 0, 1\} \quad (3)$$

から抽出した1ビットの署名データ $n_s(u)$ をつぎの手順で埋め込む。

(1)  $\Delta_j(a_1 b_1) > 0$ のとき、 $\Delta_j(b_0 a_1) \geq 1$ かつ $\Delta_j(b_1 a_2) \geq 1$ を満足するならば、

(a)  $\phi_j \text{ XOR } n_s(u) = 0$ のときそのまま、

(b)  $\phi_j \text{ XOR } n_s(u) = 1$ のとき $a_1$ を1要素分右へ移動する。

(2)  $\Delta_j(a_1 b_1) \leq 0$ のとき、 $\Delta_j(b_0 a_1) \geq -\Delta_j(a_1 b_1) + 2$ を満足するならば、

(a)  $\phi_j \text{ XOR } n_s(u) = 0$ のときそのまま、

(b)  $\phi_j \text{ XOR } n_s(u) = 1$ のとき $a_1$ を1要素分左へ移動する。上記(1)及び(2)の規約を式(4)とする。ただし、 $k_0(t_0) = 0$ のときは署名データは埋め込まない。

【0016】また、この手順により埋め込まれた署名データは、つぎの手順で容易に復号できる。

(1)  $\Delta_j(a_1 b_1) > 0$ のとき、 $\Delta_j(b_0 a_1)$

【0013】

【署名方法】

【署名文の埋め込み】送信者および送信文書の正当性を示すための署名は、デジタル化された文書画像データに署名文を直接埋め込むことにより行う。すなわち、走査線に分解された画像データから鍵でランダムに指定した走査線を参照し、この参照走査線と符号化走査線上的変化要素間の距離の偶奇性を1ビットの署名データにより変調する。

【0014】まず、符号化走査線と $s$ 本の参照走査線上的変化要素などについて以下のように定義する(図3参照)。

$a_1$ : 符号化走査線上の注目ランゲルス(以下 $R_L$ と記述する)の最初の変化要素。ただし、第1 $R_L$ は除く。

$b_1^{(j)}$ : 第 $j$ ( $1 \leq j \leq s$ )参照走査線上の $a_1$ に対応する $a_1$ と同色の変化要素。すなわち、 $a_1$ の直上要素が $a_1$ と同色のときは $a_1$ 以前の $R_L$ の最初の変化要素、 $a_1$ の直上要素が $a_1$ と逆色のときは $a_1$ より右にある $R_L$ の最初の変化要素。

$\Delta_j(a_1 b_1)$ : 変化要素 $a_1$ と $b_1^{(j)}$ の距離。  
 $\phi_j$ :  $\Delta_j(a_1 b_1)$ の偶奇性を示し、 $\Delta_j(a_1 b_1)$ が偶数ならば0、 $\Delta_j(a_1 b_1)$ が奇数ならば1とする。

【0015】このとき、 $r$ ビットの署名鍵

$0(t_0') = 1$ から $k_0(t_0) = 1$ までのビット長

1と第 $j$ 参照走査線上的変化要素 $b_1^{(j)}$ 間の距離 $\Delta_j(a_1 b_1)$ の偶奇性 $\phi_j$ に、 $q$ ビットの署名文

$\geq 1$ かつ $\Delta_j(b_1 a_2) \geq 1$ を満足するならば、  
 $n_s(u) = \phi_j$ 。

(2)  $\Delta_j(a_1 b_1) \leq 0$ のとき、 $\Delta_j(b_0 a_1) \geq -\Delta_j(a_1 b_1) + 2$ を満足するならば、  
 $n_s(u) = \phi_j$ 。

そして、抽出された1ビットの署名データ $n_s(u)$ を署名文 $N_s$ として組み立てる。上記(1)及び(2)の規約を式(5)とする。

【0017】署名文の埋め込みはこのように文書画像の構造(複数の走査線にわたる $R_L$ のパターン)に依存して行われるので、第三者が署名文を復号したり、文書画像 $M$ を偽造・改ざんすることは困難である。

【0018】

【スクランブル鍵の生成】つぎに、暗号化プロセスにおけるスクランブル鍵 $K_i$  ( $i=1, 2, \dots$ )の生成方法について示す。 $K_i$ は、直前に使用された $K_{i-1}$ (当初は署名鍵 $K_0$ )に基づき、文書画像データの構造から生

成される。まず、最大参照走査線数  $s$  について

$$s \leq 2^h \quad (6)$$

を満足する最小の整数値  $h$  を求め、これを  $h_0$  とする。系列  
直前の鍵  $K_{i-1}$  のビット系列からつぎの  $h_0$  個のビット

$$\{k_{i-1} \mid ((k_{i-1} + \alpha) \bmod r) \mid \alpha = 0, 1, \dots, h_0 - 1\} \quad (7)$$

を抽出し、これを 10進数  $d_0$  に変換する。つぎに、

$$l = (d_0 + 1) \bmod s \quad (8)$$

により参照走査線番号  $l$  を決定する。そして、符号化走査線上の変化要素  $a_l$  と第  $l$  参照走査線上の変化要素  $b_l$  返してスクランブル鍵

$l$  (1) 間の距離  $\Delta l$  ( $a_l$   $b_l$ ) の偶奇性  $\phi_l$  を求め、

$$K_l = \{k_l(t_l) \mid t_l = 0, 1, \dots, r-1; k_l(t_l) = 0, 1\} \quad (9)$$

を得る。

【0019】この報告では、このように文書画像の構造に依存して逐次生成される鍵により符号化されたデータを転置するスクランブル方法を用いる。その具体的な方法をつぎに示す。

$$C = E(K, M) \quad (10)$$

と表現する。従来のハードウェアによる転置法では  $E$  の構造が固定されているため、転置の自由度は鍵の自由度

$$C_l = E(K_l, M) \quad (i = 1, 2, \dots) \quad (11)$$

が得られるのみであった。これに対し、本手法では  $E$  をソフトウェアシステムで構成する。鍵  $K_l$  を入力すると、システムは自動的に相異なる部分暗号器 (復号器)

$$C = E_{l1}(K_{l1}, E_{l2}(K_{l2}, \dots, E_{ln}(K_{ln}, M) \dots)) \quad (12)$$

を出力する。

【0021】このシステムでは式 (12) を以下の手順で実現する。まず、スクランブル鍵  $K_l$  ( $l = 1, 2, \dots$ ) が入力されると、その一部を利用してメモリ上に  $E_{l1}$  および  $K_{l1}$  を展開する。すなわち、図 4 に示すように鍵  $K_l$  のビット系列の一部が機械語プログラムとして利用され、引き続き系列が部分鍵として用いられる。つぎに、メモリ上に格納された符号データを 2 のべき乗単位にブロック化して入力し、生成したプログラムによりブロック内の個々のデータの座標値を指数としてスクランブルする。その演算結果を新しい座標値として符号データを出力する。この方法を 1 ブロックの転置が終了すると、暗号化系列と共につぎのブロックのためのスクランブル鍵  $K_{l+1}$  が出力され、 $K_{l+1}$  に基づいて同様な処理が繰り返される。この処理の流れを図 5 に示す。

【0022】さて、ここで用いられる  $E_{lj}$  は、全体でブロック転置暗号を構成する必要性から、各々が全単射 (1:1) の写像関数でなければならない。そこで、本

$$C = \text{XOR}(K_{\text{XOR}}, M) \quad (13)$$

と表す。明らかに

$$M = \text{XOR}(K_{\text{XOR}}, C) \quad (14)$$

であり、この演算では、式 (13) あるいは式 (14) の関係にある 2 つの座標値  $M$ 、 $C$  をもつ各々のデータが相互に置き換えられる。すなわち、ブロック内のすべて

【0020】

【スクランブル方法】

【暗号化の概念】 平文  $M$  を鍵  $K$  で暗号化して得られる暗号文を

にすべてを依存し、

$E_{lj}$  および部分鍵  $K_{lj}$  ( $l = 1, 2, \dots; j = 1, 2, \dots, n$ ) を生成・配列し、さらにそれらを結合して

システムでは、中央処理装置 (CPU) が持つ基本演算命令とその拡張演算を各  $E_{lj}$  として選定する。これにより、暗号器および復号器はわずかのメモリ占有量で高速な転置処理を行なうことができる。

【0023】

【全単射写像関数となる CPU 命令】 CPU が持つ加減乗除演算等の命令の中で被演算数と演算結果との間に全単射写像を構成するものは限られている。以下にここで用いた CPU 命令を示す。ただし、上述のスクランブルを生成する CPU のレジスタ長を  $R$  とする。したがって、このレジスタでアドレッシング可能な範囲は  $0 \sim 2^R - 1$  であり、入力する 1 ブロックはこの大きさを単位とする。

【0024】

【排他的論理和】 レジスタ内の各ビットを用意された定数 (暗号化鍵) とビットごとに排他的論理和演算を行なう処理は全単射である。以下この演算を  $\text{XOR}$  で示し、被演算数を  $M$ 、鍵定数を  $K_{\text{XOR}}$  としたとき演算結果を

$$C = \text{XOR}(K_{\text{XOR}}, M) \quad (13)$$

のデータは上記の関係を満たす対で置換される。

【0025】

【左右ローテイト】 レジスタ内の各ビットを指定された

方向へ指定されたビット数だけシフトし、レジスタからはみ出した桁を逆側から入力するローテイト演算は全単射となる。以下左ローテイトをROL、右ローテイトをRORで示す。

【0026】

【加減算】レジスタ内のデータに、用意された定数を加

$$C = \text{ADD}(K_{\text{ADD}}, M) \\ = M + K_{\text{ADD}} \bmod 2^R$$

となる。すなわち、個々のデータは距離 $K_{\text{ADD}}$ だけ平行移動される。上で述べた全単射写像を構成するCPU命令の他、以下のような拡張演算も全単射写像となり、転置効果を向上させる。

【0027】

【ビット位置の入れ換え】レジスタ内の各ビットを、特定の位置へ入れ換える処理は全単射である。以下この演算をEXT1で示す。鍵 $\text{EXT1}(i)$  ( $i=0, 1, \dots, R-1$ )はレジスタ長 $R$ に対応した $R$ 個の要素からなる配列であり、それぞれの値は移動先のビット位置を示す。

【0028】

【部分空間ごとの位置】レジスタ長 $R$ で表現される全アドレス空間を互いに交わらない部分空間に分割し、各部分空間ごとに全単射演算するとき、全体の転置も全単射である。部分空間への分割方法としては、レジスタ長を短くする方法、レジスタ内に含まれるビット1の個数による分類法がある。ここでは、後者の部分空間を利用し、この部分空間では演算可能なROLを適用してEXT2で示す。ただし、レジスタ内のすべてがビット1または0のときを除いた部分空間に適用する。従って、鍵 $\text{EXT2}(i)$  ( $i=0, 1, \dots, R-2$ )はレジスタ長 $R$ のとき要素数 $R-1$ 個の配列で、その第 $i$ 要素はレジスタ内にビット1が $i$ 個存在する部分空間に適用するROLの鍵を示す。

【0029】

【暗号器・復号器】式(12)で与えられる暗号器および復号器は、鍵 $K_i$  ( $i=1, 2, \dots$ )のビット系列から機械語命令とそのオペランドを生成する。すなわち、 $K_i$ から2または3ビットを入力し、表1の変換表に基づいて $n$ 段の命令群を生成し、部分暗号器 $E_{ij}$  ( $j=1, 2, \dots, n$ )の系列として主記憶のコード領域に展開する。このとき、同時に各転置演算が必要となるオペランド、すなわち部分鍵 $K_{ij}$  ( $j=1, 2, \dots, n$ )も $K_i$ のビット系列から得る。ただし、暗号器の段数 $n$ は予め設定しておくものとする。たとえば、レジスタ長を $R=16$ (ビット)とすると、各々の演算で必要とされる部分鍵のビット数は表2となる。

【表1】

減算する処理は全単射である。以下、加算をADD、減算をSUBで示す。一般にADDのような算術演算命令は、その結果が桁あふれした場合、フラッグレジスタが影響を受けるが、この点を無視してレジスタ内のデータのみに着目すると

(15)

変換表

入力ビット	出力コード
110	XOR
000	ROL
001	ROR
010	ADD
011	SUB
111	EXT1
10	EXT2

【表2】

演算で必要とされる部分鍵のビット数

コード	所要ビット数
XOR	16
ROL	4
ROR	4
ADD	16
SUB	16
EXT1	64 (=4×16)
EXT2	60 (=4×15)

【0030】

【署名および暗号化の具体的手順】本実施例において送受信間で秘匿すべきデータは $(s, n, K_0)$ である。以下にその具体的手順を示す。

【0031】

【署名プロセス】Step1: 文書画像 $M_i$ 、署名鍵 $K_0$ および署名文 $N_s$ を入力すると共に $t_0 \leftarrow 0$ 、 $t_1 \leftarrow 0$ 、 $u \leftarrow 0$ とする。

Step2:  $s$ 本の走査線をメモリに蓄積すると共に符号化・伝送する。

Step3: Step4からStep9を署名文 $N_s$ の埋め込みが終了するまで繰り返す。

Step4: Step5からStep8を符号化走査線の走査が終了するまで繰り返す。

Step5: 変換要素 $a_1$ を検出する。

Step6:  $K_0$ から $k_0(t_0)$ を含む $h_0$ ビットを抽出して10進数 $d_0$ に変換し、式(8)により $l$ を決定する。第1参照走査線上の変換要素 $b_1^{(1)}$ を検出し、 $k_1(t_1) \leftarrow d_1$ とする。 $t_1 \leftarrow t_1 + 1$ とする。

Step7:  $k_0(t_0) = 1$ ならば、直前の $k_0(t_0') = 1$ から $k_0(t_0) = 1$ までのビット長 $j$ を式(2)により決定する。第 $j$ 参照走査線上の変換要素 $b_1^{(j)}$ を検出して式(4)で署名データ $n_s(u)$ を埋め込む。 $n_s(u)$ が埋め込まれたとき $u \leftarrow u + 1$ とする。

Step 8:  $t_0 \leftarrow (t_0 + 1) \bmod r$  とする。  
 Step 9: 符号化走査線を符号化・伝送すると共に、符号化走査線を第1参照走査線に、第v参照走査線を第v+1 ( $v=1, 2, \dots, s-1$ ) 参照走査線に更新する。

Step 10: 暗号化プロセスに移行する。

【0032】

【暗号化プロセス】Step 1:  $i=1$  とし、 $n$  を入力する。

Step 2: Step 3 から Step 8 をすべての符号データについて終了するまで繰り返す。

Step 3:  $t_i \leftarrow 0$ ,  $t_{i+1} \leftarrow 0$  とする。

Step 4: Step 5 から Step 8 を符号データ量が  $2^R$  ビット以上になるまで繰り返す。

Step 5: Step 6 から Step 7 を符号化走査線の走査が終了するまで繰り返す。

Step 6: 変換要素  $a_i$  を検出する。

Step 7:  $K_i$  から  $k_i(t_i)$  を含む  $h_0$  ビットを抽出して10進数  $d_0$  に変換し、式 (8) により  $i$  を決定する。第1参照走査線上的変換要素  $b_i^{(1)}$  を抽出し、  
 $k_{i+1}(t_{i+1}) = \phi i$  とする。 $t_i = (t_i + 1) \bmod r$ ,  $t_{i+1} = (t_{i+1} + 1) \bmod r$  とする。

Step 8: 符号化走査線を符号化すると共に、符号化走査線を第1参照走査線に、第v参照走査線を第v+1 ( $v=1, 2, \dots, s-1$ ) 参照走査線に更新する。

Step 9:  $K_i$  から  $n$  段の転置暗号器を主記憶のコード領域に展開する。

Step 10: アドレス  $m = (0 \sim 2^R - 1)$  上の符号データを生成された暗号器により別途用意されたバッファメモリ上のアドレス  $m' = (0 \sim 2^R - 1)$  に転置し、アドレス0から順に出力・伝送する。 $i \leftarrow i + 1$  とする。

【0033】

【認証プロセス】Step 1: 受信データ  $F(M_i)$ 、認証鍵  $K_0$  を入力すると共に  $t_0 \leftarrow 0$ ,  $t_1 \leftarrow 0$ ,  $u \leftarrow 0$  とする。

Step 2: 符号データを  $a$  走査線分復号してメモリに蓄積し、参照走査線とするとする。

Step 3: Step 4 から Step 10 を署名文  $N_s$  の復号が終了するまで繰り返す。

Step 4: 符号データを1走査線分復号し、符号化走査線とする。

Step 5: Step 6 から Step 9 を符号化走査線の走査が終了するまで繰り返す。

Step 6: 変換要素  $a_i$  を検出する。

Step 7:  $K_0$  から  $k_0(t_0)$  を含む  $h_0$  ビットを抽出して10進数  $d_0$  に変換し、式 (8) により  $i$  を決定する。第1参照走査線上的変換要素  $b_i^{(1)}$  を抽出し、  
 $k_i(t_i) \leftarrow \phi i$  とする。 $t_i \leftarrow t_i + 1$  とする。

Step 8:  $k_0(t_0) = 1$  ならば、直前の  $k_0(t$

$0')$  = 1 から  $k_0(t_0) = 1$  までのビット長  $j$  を式 (2) により決定する。第j参照走査線上的変換要素  $b_j^{(1)}$  を抽出して式 (5) で署名データ  $n_s(u)$  を抽出する。 $n_s(u)$  が抽出されたとき  $u \leftarrow u + 1$  とする。

Step 9:  $t_0 \leftarrow (t_0 + 1) \bmod r$  とする。

Step 10: 符号化走査線を第1参照走査線に、第v参照走査線を第v+1 ( $v=1, 2, \dots, s-1$ ) 参照走査線に更新する。

Step 11:  $N_s$  が正当ならば復号プロセスに移行し、そうでなければ、受信データ  $F(M_r)$  を棄却する。

【0034】

【復号プロセス】Step 1:  $i=1$  とし、 $n$  を入力する。

Step 2: Step 3 から Step 10 をすべての符号データについて終了するまで繰り返す。

Step 3: 受信データ  $F(M_i)$  から  $2^R$  ビット抽出すると共に  $t_i \leftarrow 0$ ,  $t_{i+1} \leftarrow 0$  とする。

Step 4: 鍵  $K_i$  から  $n$  段の転置暗号器を主記憶のコード領域に展開する。

Step 5: アドレス  $m = (0 \sim 2^R - 1)$  上の受信データを生成された復号器により別途用意されたバッファメモリ上のアドレス  $m' = (0 \sim 2^R - 1)$  に転置し、アドレス0から順に出力する。 $i \leftarrow i + 1$  とする。

Step 6: 復号された符号データを画像データに変換し、変換された走査線について Step 7 から Step 10 を繰り返す。

Step 7: Step 8 から Step 9 を符号化走査線の走査が終了するまで繰り返す。

Step 8: 変換要素  $a_i$  を検出する。

Step 9:  $K_i$  から  $k_i(t_i)$  を含む  $h_0$  ビットを抽出して10進数  $d_0$  に変換し、式 (8) により  $i$  を決定する。第1参照走査線上的変換要素  $b_i^{(1)}$  を抽出し、  
 $k_{i+1}(t_{i+1}) = \phi i$  とする。 $t_i = (t_i + 1) \bmod r$ ,  $t_{i+1} = (t_{i+1} + 1) \bmod r$  とする。

Step 10: 符号化走査線を第1参照走査線に、第v参照走査線を第v+1 ( $v=1, 2, \dots, s-1$ ) 参照走査線に更新する。

【0035】

【スクランブル効果の検討】前述のスクランブル手順の実行結果として出力された符号データの0, 1ビットの系列が、十分にランダムであるかを検討するため、すでに十分な擬ランダム性が証明されているフィードバックシフトレジスタによるM系列の数値列と以下の項目についてその性質を比較、検討する。ただし、ここではファクシミリ文書の符号化方式としてMH方式を、入力文書としてCCITTのテストチャートNo. 4を用いている。

【0036】(1) 連の個数の生起確率

連とは2値系列上の1または0のp個の並びであり、その間隔は0または1である。このとき、M系列では1ま

$$[\text{長さ } p \text{ の連の個数}] / [\text{長さ } p + 1 \text{ の連の個数}] = 2 \quad (16)$$

が示される。ただし、q段シフトレジスタ上では、長さqおよびq-1の場合を除いてこの条件を満足する。そこで、MH方式による符号データおよびそれをレジスタ長Rを16ビット、基本命令の段数nを32としてスクランブルしたデータについて連の状態を調べた。図6はそれぞれの場合における連の個数を2を底とした対数でグラフ化した結果を示す。ただし、点線は等価段数(16段)のシフトレジスタによるものである。この結果、

$$C_0(h) = \lim_{j \rightarrow \infty} \frac{1}{2j+1} \sum_{r=-j}^{r=j} y_r y_{r+h} \quad (17)$$

により調査した<sup>(11)</sup>。ただし、 $\{y_r\}$ は2値系列上の項を、hは2項 $\{y_r\}$ と $\{y_{r+h}\}$ 間の距離をそれ

$$\begin{aligned} h \neq 0 \text{ ならば } C_0(h) &= 0 \\ h = 0 \text{ ならば } C_0(h) &= 1 \end{aligned}$$

により系列のランダム性が特徴づけられる。そこで、同様にMH方式による符号データおよびそれをレジスタ長Rを16ビット、段数nを32としてスクランブルしたデータについてそれぞれの自己相関関数を求めると図7を得る。この結果、本手法の演算結果は点線で示すシフトレジスタの自己相関関数と同様、式(18)をほぼ満たしていることがわかる。

【0038】

【発明の効果】以上に実施例を挙げて詳しく説明したように、本発明のファクシミリにおいては、①送信者および送信文書の正当性を示すための署名文を画像データの一部に埋め込むことにより署名を施し、②符号化された画像データに対してスクランブル処理をする。このような手法の採用により、本発明により実現される拡張ボード型のFAXアダプターはハードウェア的に簡易であることから低価格であり、従って、機密保護専用アダプターと比較して低コストでセキュリティ通信を実現できる。また、スクランブル処理は符号データを2のべき乗単位にブロック化して転置暗号化するために、ファクシミリのMH、MRおよびMMRの各種符号化方式や伝送符号

または0の連の個数について、

本手法によりスクランブルされたデータ系列は、連の個数に関してシフトレジスタの場合に極めて優れた結果が得られている。nを増加させるとさらに良好な結果が確かめられる。

【0037】(2) 自己相関関数

つぎに、スクランブルされたデータ系列のランダム性を次式で定義される離散的自己相関関数

【数1】

それ示し、

(18)

量などに関する制約を受けない利点がある。また、本発明のファクシミリは、現在実用化されているカードFAXシステムにおけるアプリケーション・ソフトウェアを一部変更・追加するだけで実現可能であり、機密保護専用のアダプタよりも低コストでセキュリティ通信を実現できる。

【図面の簡単な説明】

【図1】従来のカードFAXの構成例を示す図。

【図2】本発明の一実施例における機密保護システムを示す図。

【図3】符号化定数線と本発明の参照定数線との変遷図表の定義を示す図。

【図4】部分暗号器(復号器)E<sub>11</sub>および部分鍵K<sub>11</sub>を示す図。

【図5】本発明の実施例におけるスクランブル処理の流れを示す図。

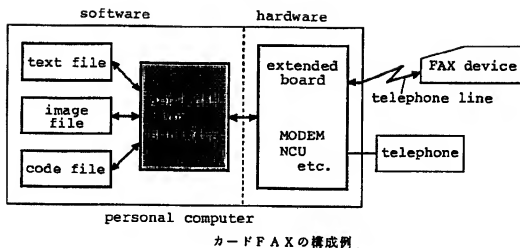
【図6】本発明の実施例における連の個数を2を底とした対数でグラフ化して示す図。

【図7】自己相関関数を示す図。

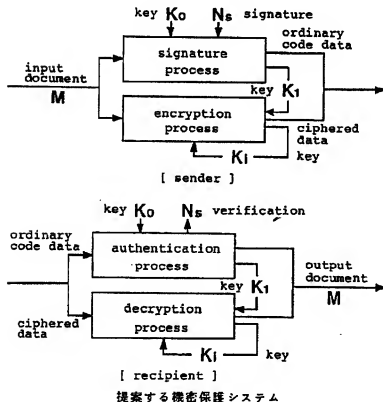
【図4】



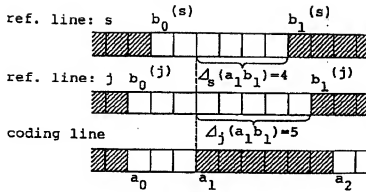
【図1】



【図2】

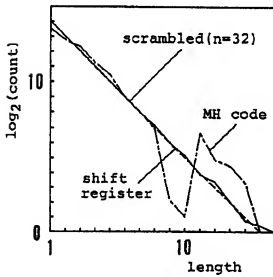


【図3】



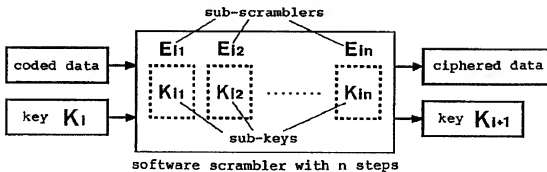
変化要素の定義

【図6】



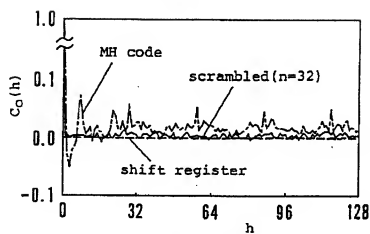
連の個数

【図5】



スクランブル処理の流れ

【図7】



自己相関関数